BIAX Corporation v. Intel
Civil Action No. 2:05-cv-184-TJW

EXHIBIT 1
(PART 1)
FIRST AMENDED COMPLAINT FOR PATENT INFRINGEMENT

United States Patent [19]

Morrison et al.

[11] **Patent Number: 4,847,755**

[45] **Date of Patent: Jul. 11, 1989**

[54] **PARALLEL PROCESSING METHOD AND APPARATUS FOR INCREASING PROCESSING THROUGHOUT BY PARALLEL PROCESSING LOW LEVEL INSTRUCTIONS HAVING NATURAL CONCURRENCIES**

[75] Inventors: **Gordon E. Morrison**, Denver; **Christopher B. Brooks**; **Frederick G. Gluck**, both of Boulder, all of Colo.

[73] Assignee: **MCC Development, Ltd.**, Boulder, Colo.

[21] Appl. No.: **794,221**

[22] Filed: **Oct. 31, 1985**

[51] **Int. Cl.**[4] **G06F 15/16**; G06F 13/00

[52] **U.S. Cl.** **364/200**; 364/229

[58] **Field of Search** ... 364/200 MS File, 900 MS File

[56] **References Cited**

U.S. PATENT DOCUMENTS

3,343,135	9/1967	Freiman et al.	340/172.5
3,611,306	10/1971	Reigel	340/172.5
3,771,141	11/1973	Culler	340/172.5
4,104,720	8/1978	Gruner	364/200
4,109,311	8/1978	Blum et al.	364/200
4,153,932	5/1979	Dennis et al.	364/200
4,181,936	1/1980	Kober	364/200
4,228,495	10/1980	Bernhard	364/101
4,229,790	10/1980	Gilliland et al.	364/200
4,241,398	12/1980	Caril	364/200
4,270,167	5/1981	Koehler et al.	364/200
4,430,707	2/1984	Kim	364/200
4,435,758	3/1984	Lorie et al.	364/200
4,466,061	8/1984	DeSantis	364/200
4,468,736	8/1984	DeSantis	364/200
4,514,807	4/1985	Nogi	364/200
4,574,348	3/1986	Scallon	364/200

OTHER PUBLICATIONS

Dennis, "Data Flow Supercomputers", Computer, Nov., 1980, pp. 48–56.

Hagiwara, H. et al., "A Dynamically Microprogrammable, Local Host Computer with Low-Level Parallelism", IEEE Transactions on Computers, C-29, No. 7, Jul., 1980, pp. 577–594.

Fisher et al., "Microcode Compaction: Looking Backward and Looking Forward", National Computer Conference, 1981, pp. 95–102.

Fisher et al., "Using an Oracle to Measure Potential Parallelism in Single Instruction Stream Programs", IEEE No. 0194-1895/0000/0171, 14th Annual Microprogramming Workshop, Sigmicro, Oct., 1981, pp. 171–182.

J. R. Vanaken et al., "The Expression Processor", IEEE Transactions on Computers, C-30, No. 8, Aug., 1981, pp. 525–536.

Bernhard, "Computing at the Speed Limit", IEEE Spectrum, Jul., 1982, pp. 26–31.

Davis, "Computer Architecture", IEEE Spectrum, Nov., 1983, pp. 94–99.

Hagiwara, H. et al., "A User-Microprogrammable Local Host Computer with Low-Level Parallelism", Article, Association for Computing Machinery, #0149-7111/83/0000/0151, 1983, pp. 151–157.

McDowell, Charles Edward, "SIMAC: A Multiple ALU Computer", Dissertation Thesis, University of California, San Diego, 1983, (111 pages).

McDowell, Charles E., "A Simple Architecture for Low Level Parallelism", Proceedings of 1983 International Conference on Parallel Processing, pp. 472–477.

Requa, et al., "The Piecewise Data Flow Architecture: Architectual Concepts", IEEE Transactions on Computers, vol. C-32, No. 5, May, 1983, pp. 425–438.

Fisher, A. T., "The VLIW Machine: A Multiprocessor for Compiling Scientific Code", Computer, 1984, pp. 45–52.

Fisher et al., "Measuring the Parallelism Available for Very Long Instruction, Word Architectures", IEEE Transactions on Computers, vol. C-33, No. 11, Nov., 1984, pp. 968–976.

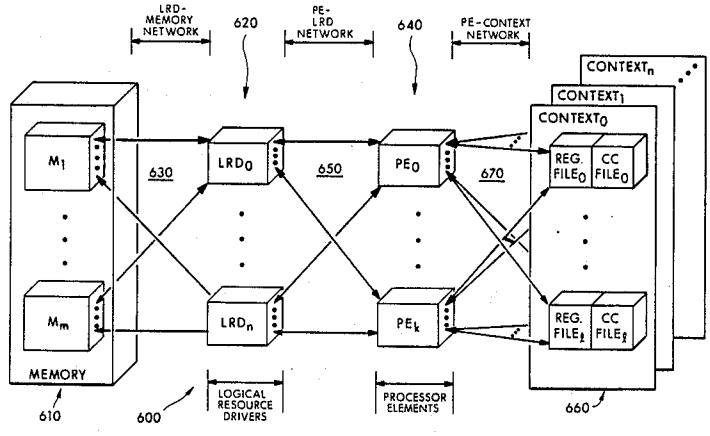
Primary Examiner—Eddie P. Chan

Attorney, Agent, or Firm—Hale and Dorr

[57] **ABSTRACT**

A computer processing system containing a plurality of processor elements operates on a statically compiled program which, based upon detected natural concurrencies in the basic blocks of the programs, includes intelligence regarding logical processor allocation and an instruction firing time in the instruction stream. Each processor element, in one embodiment, is context free and is capable of executing instructions on a per instruction basis so that dependent instructions can execute on the same or different processor elements. A processor element is capable of executing an instruction from one context followed by an instruction from another context through use of shared storage resources.

37 Claims, 17 Drawing Sheets



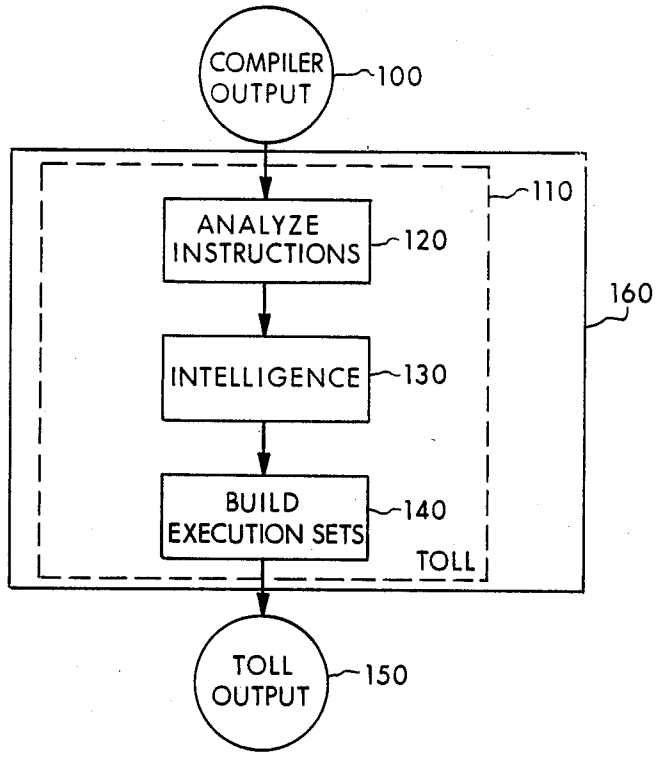


Fig. 1

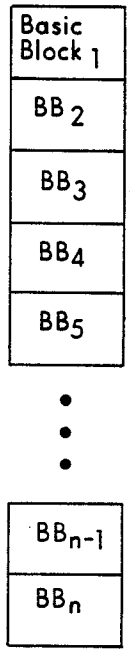


Fig. 2
Prior Art

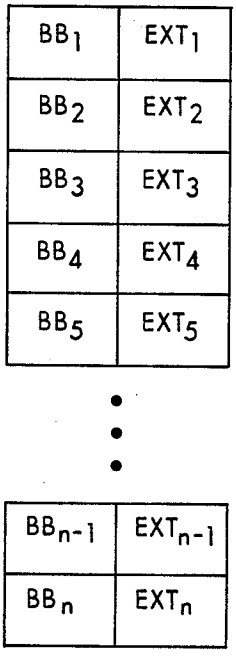


Fig. 3

Fig. 4

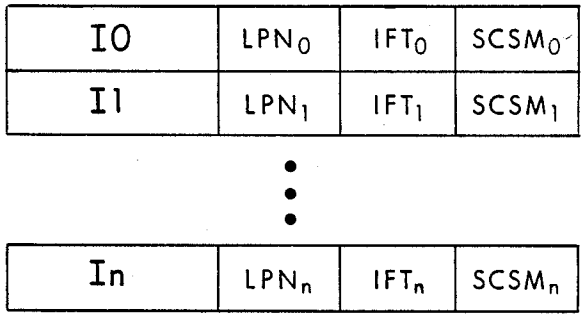
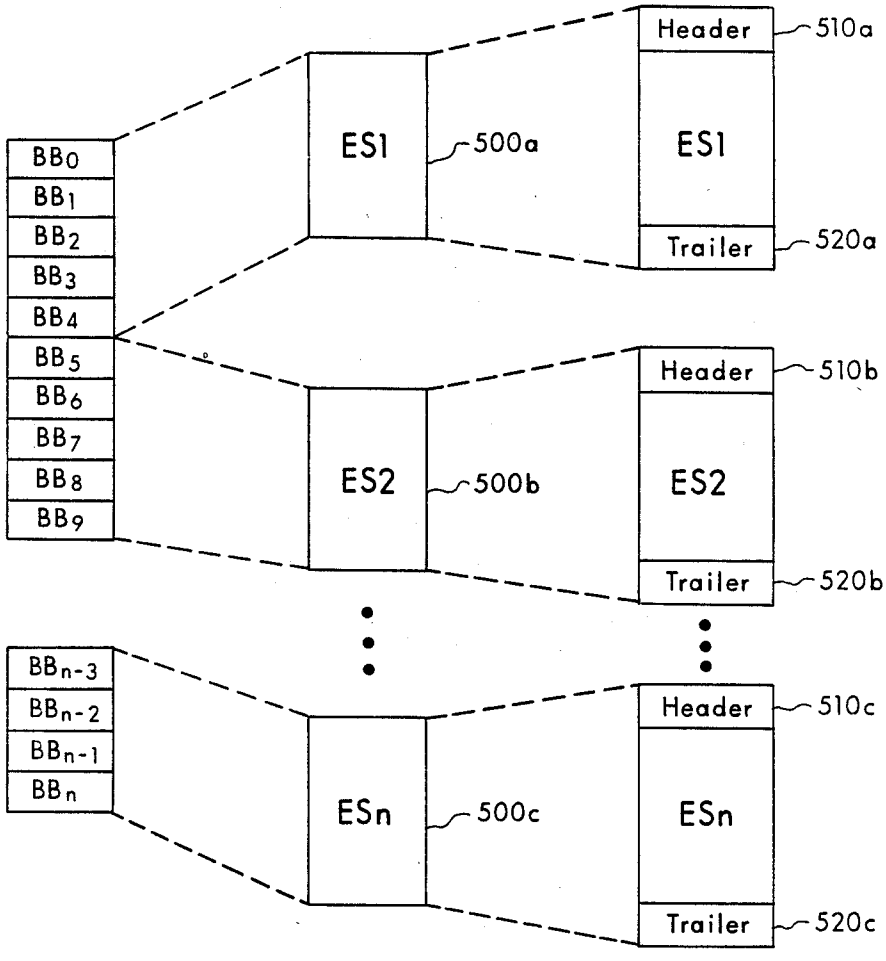

IO
LPN0
IFT0
SCSM0
I1
LPN1
IFT1
SCSM1
In
LPNn
IFTn
SCSMn

Fig. 5


BB0
BB1
BB2
BB3
BB4
BB5
BB6
BB7
BB8
BB9
BBn-3
BBn-2
BBn-1
BBn
ES1
500a
ES2
500b
ESn
500c
Header
ES1
Trailer
510a
520a
Header
ES2
Trailer
510b
520b
Header
ESn
Trailer
510c
520c

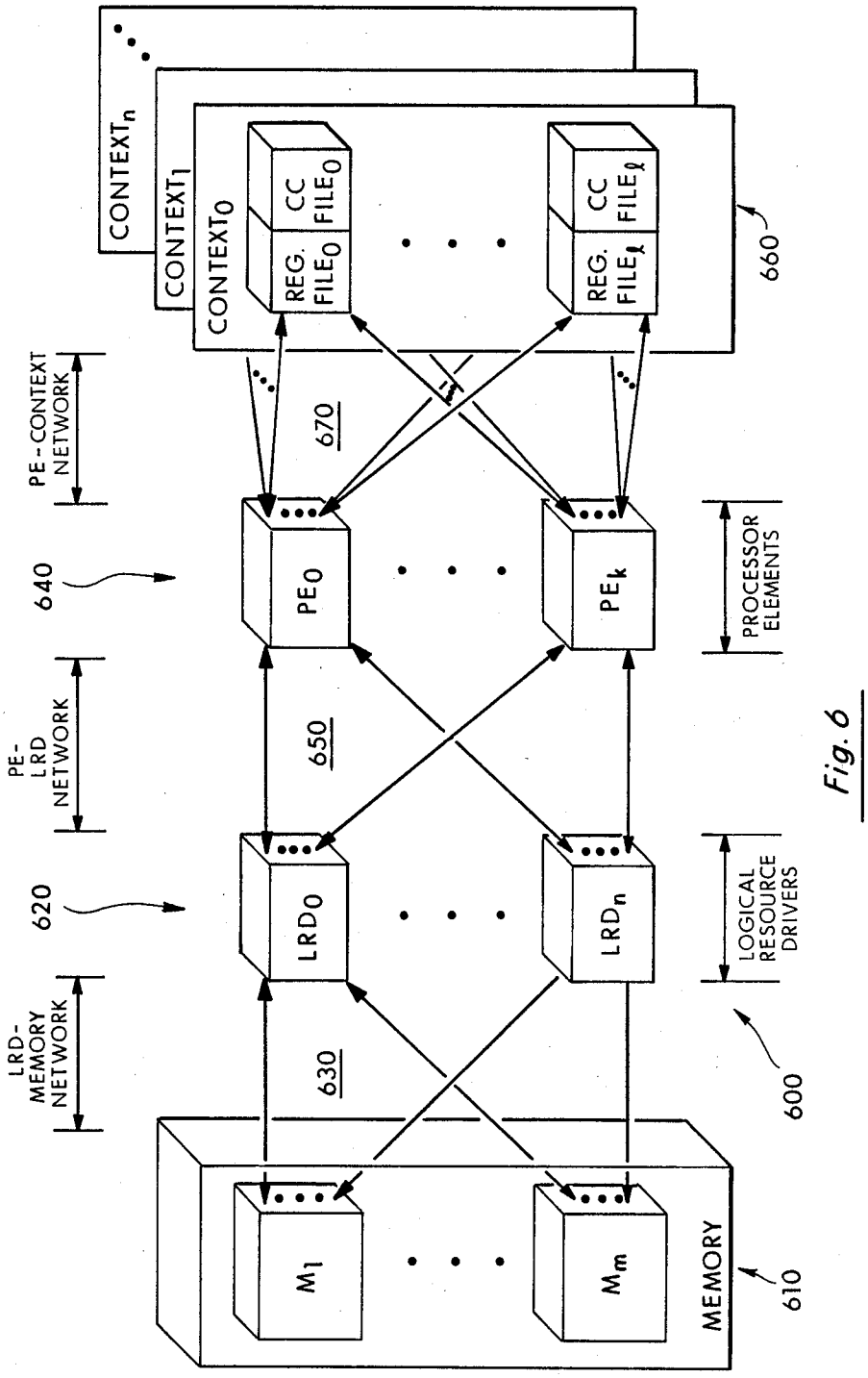

CONTEXT$_n$
CONTEXT$_1$
CONTEXT$_0$
REG. FILE$_0$
CC FILE$_0$
REG. FILE$_\ell$
CC FILE$_\ell$
660
670
PE-CONTEXT NETWORK
640
PE$_0$
PE$_k$
PROCESSOR ELEMENTS
650
PE-LRD NETWORK
620
LRD$_0$
LRD$_n$
LOGICAL RESOURCE DRIVERS
630
LRD-MEMORY NETWORK
600
M_1
M_m
MEMORY
610

Fig. 6

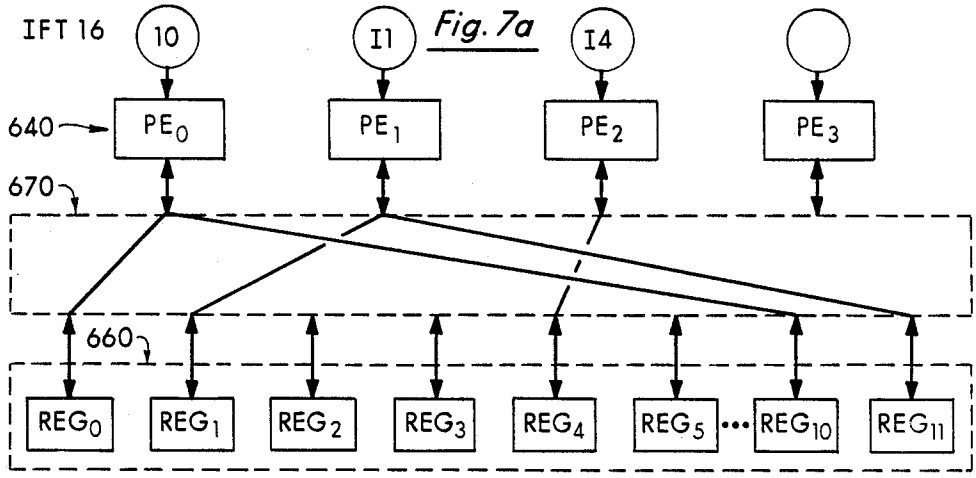


Fig. 7a

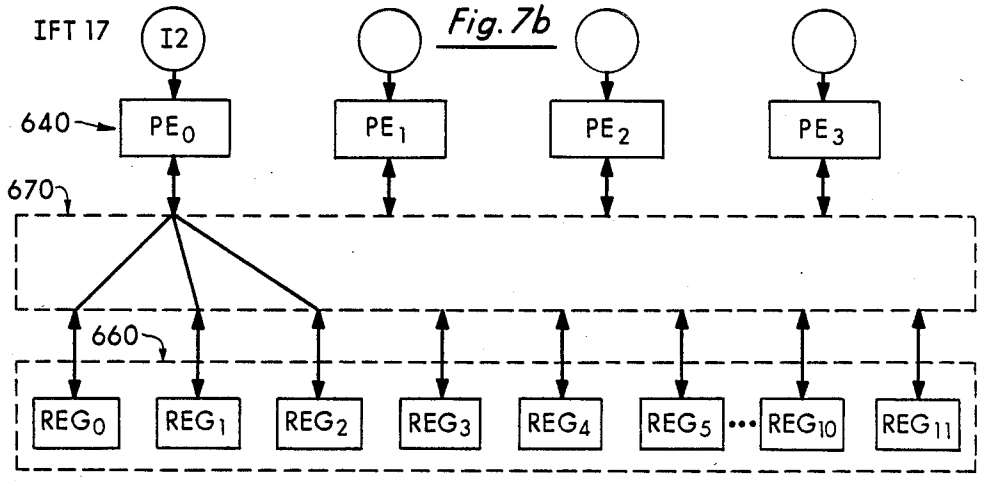


Fig. 7b

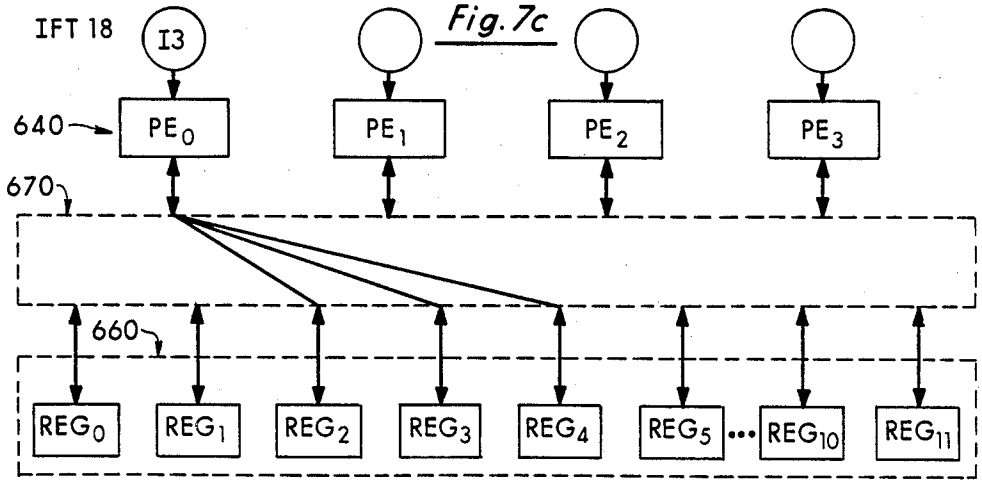


Fig. 7c

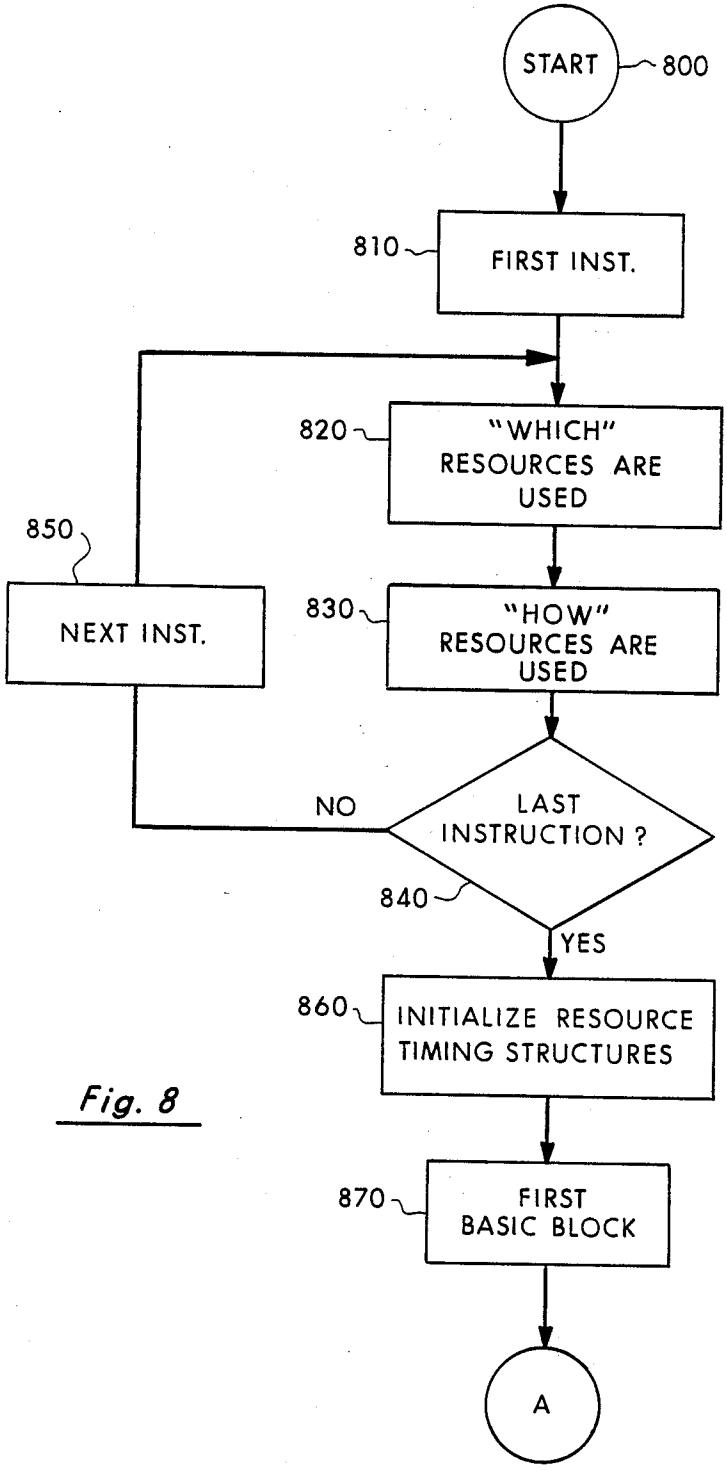

START
800
810
FIRST INST.
820
"WHICH" RESOURCES ARE USED
850
NEXT INST.
830
"HOW" RESOURCES ARE USED
NO
LAST INSTRUCTION ?
840
YES
860
INITIALIZE RESOURCE TIMING STRUCTURES
870
FIRST BASIC BLOCK
A

Fig. 8

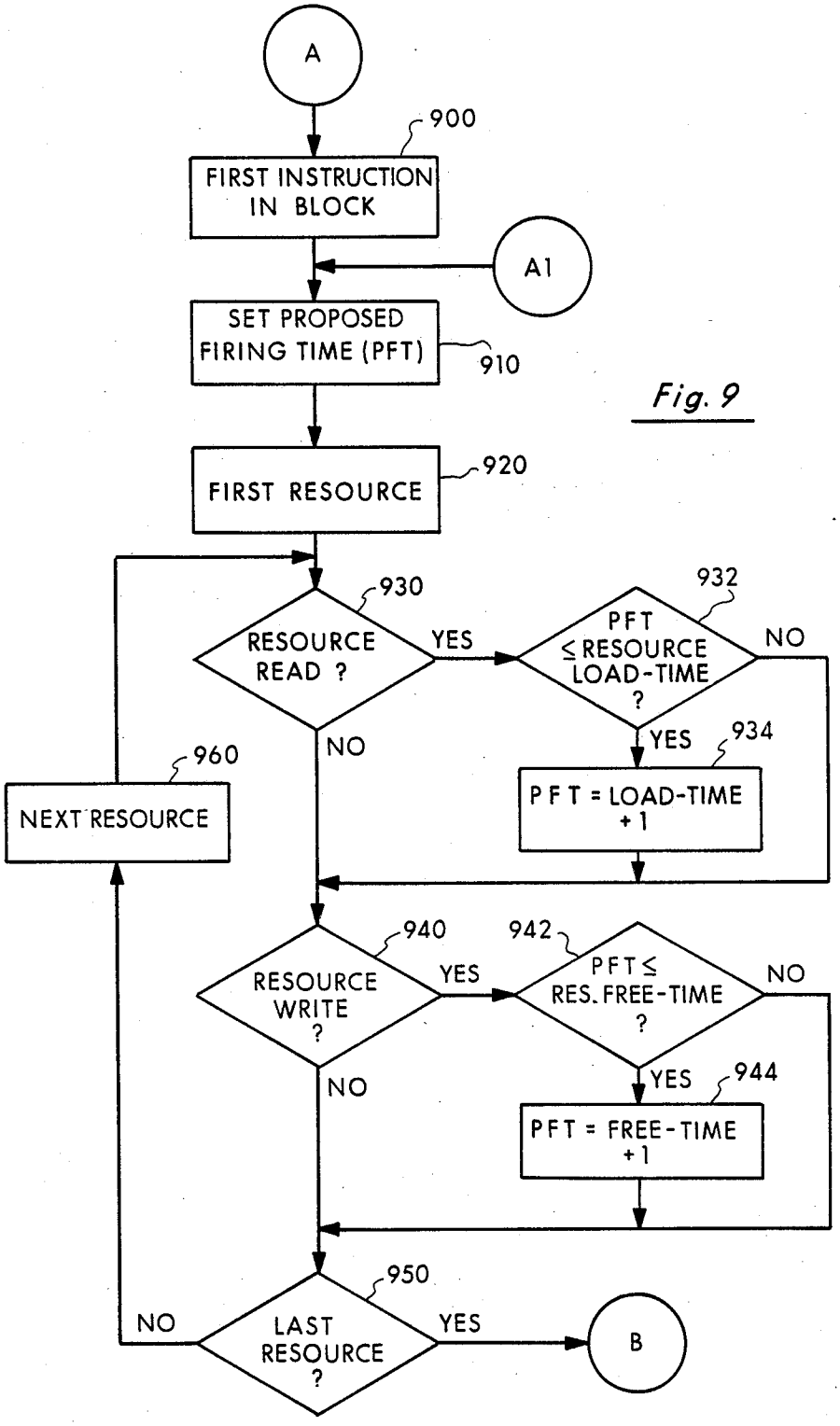

A
900
FIRST INSTRUCTION IN BLOCK
A1
SET PROPOSED FIRING TIME (PFT)
910
FIRST RESOURCE
920
930
RESOURCE READ ?
YES
932
PFT ≤ RESOURCE LOAD-TIME ?
NO
YES
934
PFT = LOAD-TIME +1
NO
960
NEXT RESOURCE
940
RESOURCE WRITE ?
YES
942
PFT ≤ RES. FREE-TIME ?
NO
YES
944
PFT = FREE-TIME +1
NO
950
LAST RESOURCE ?
NO
YES
B

Fig. 9

Fig. 10

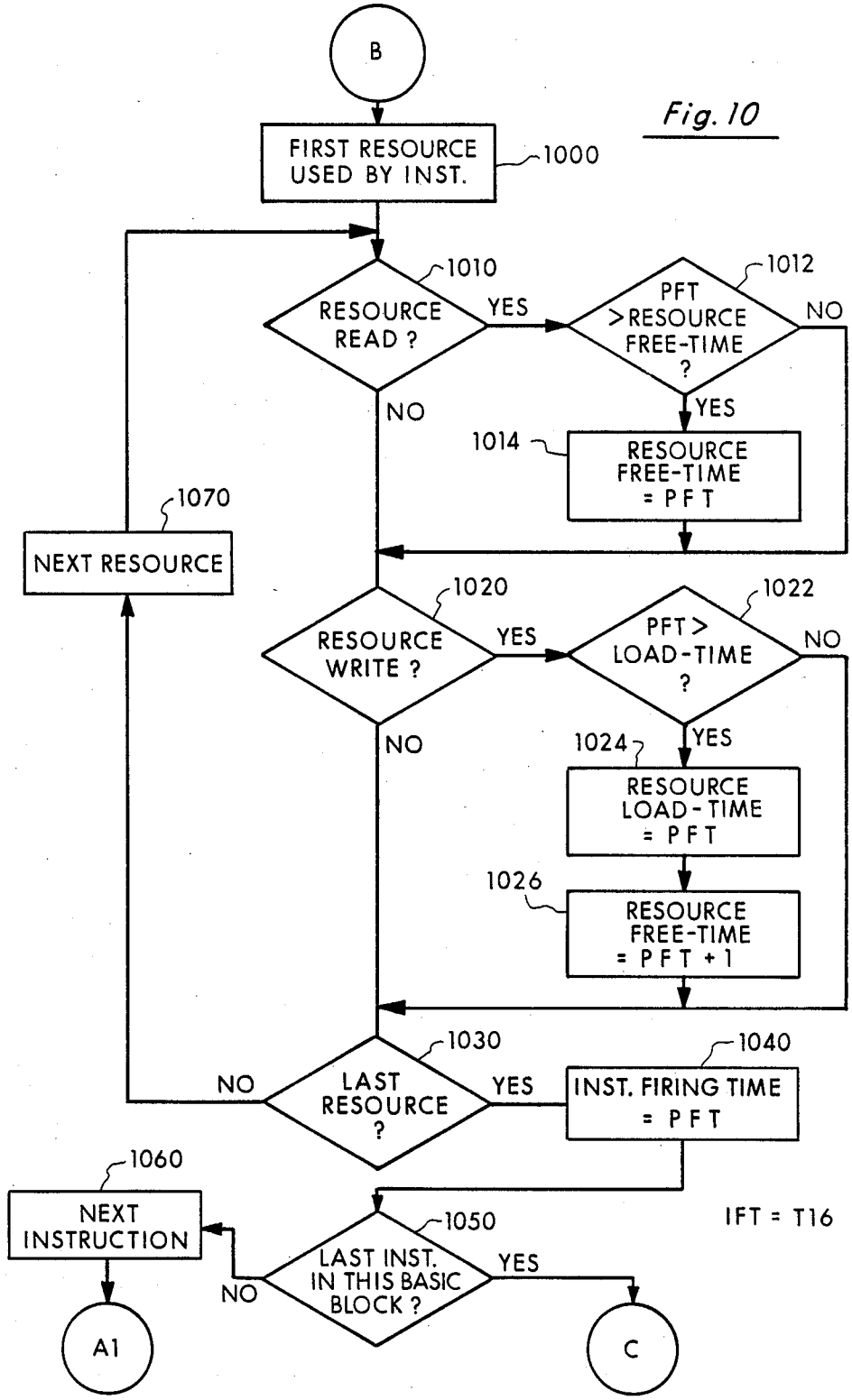
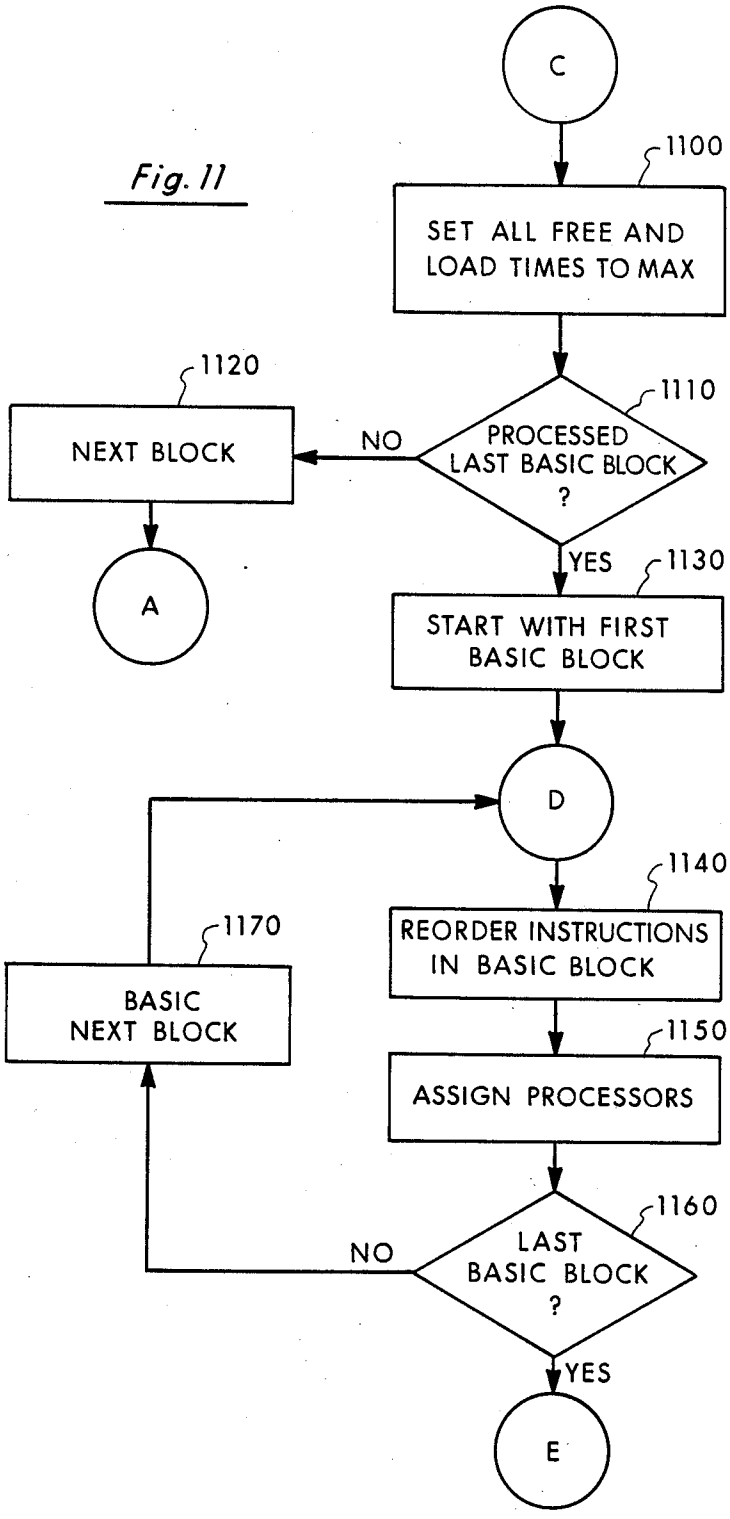

B
FIRST RESOURCE USED BY INST.
1000
1010
RESOURCE READ ?
YES
1012
PFT > RESOURCE FREE-TIME ?
NO
YES
NO
1014
RESOURCE FREE-TIME = PFT
1070
NEXT RESOURCE
1020
RESOURCE WRITE ?
YES
1022
PFT > LOAD-TIME ?
NO
NO
1024
YES
RESOURCE LOAD-TIME = PFT
1026
RESOURCE FREE-TIME = PFT + 1
1030
LAST RESOURCE ?
NO
YES
1040
INST. FIRING TIME = PFT
1060
NEXT INSTRUCTION
1050
LAST INST. IN THIS BASIC BLOCK ?
NO
YES
IFT = T16
A1
C

Fig. 11


C
1100
SET ALL FREE AND LOAD TIMES TO MAX
1120
NEXT BLOCK
NO
1110
PROCESSED LAST BASIC BLOCK ?
A
YES
1130
START WITH FIRST BASIC BLOCK
D
1140
REORDER INSTRUCTIONS IN BASIC BLOCK
1170
BASIC NEXT BLOCK
1150
ASSIGN PROCESSORS
1160
NO
LAST BASIC BLOCK ?
YES
E

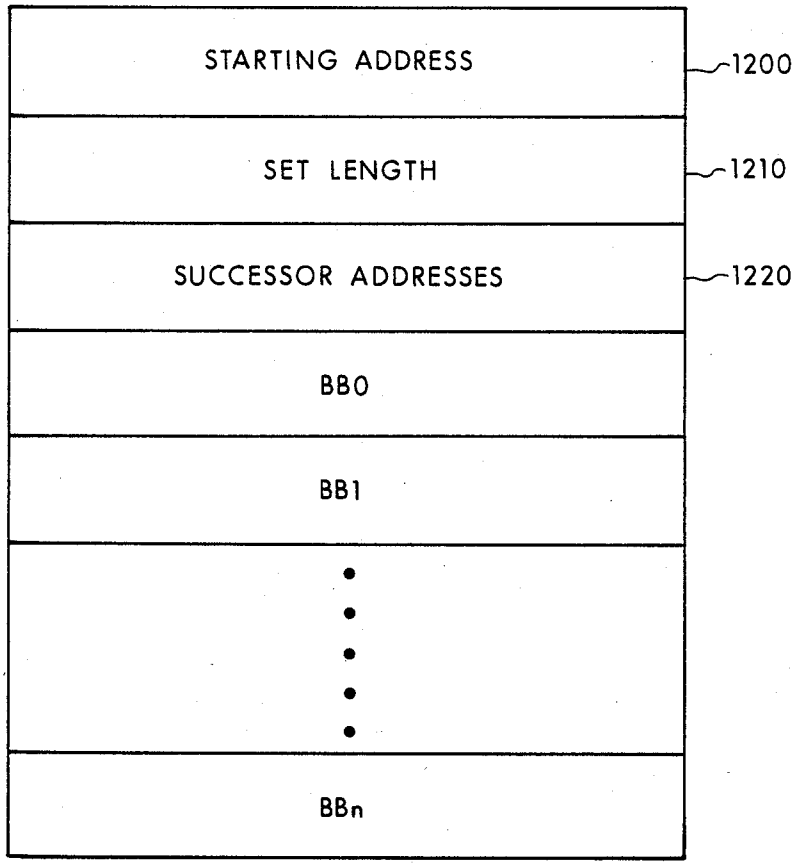

STARTING ADDRESS
1200
SET LENGTH
1210
SUCCESSOR ADDRESSES
1220
BB0
BB1
BBn

Fig. 12

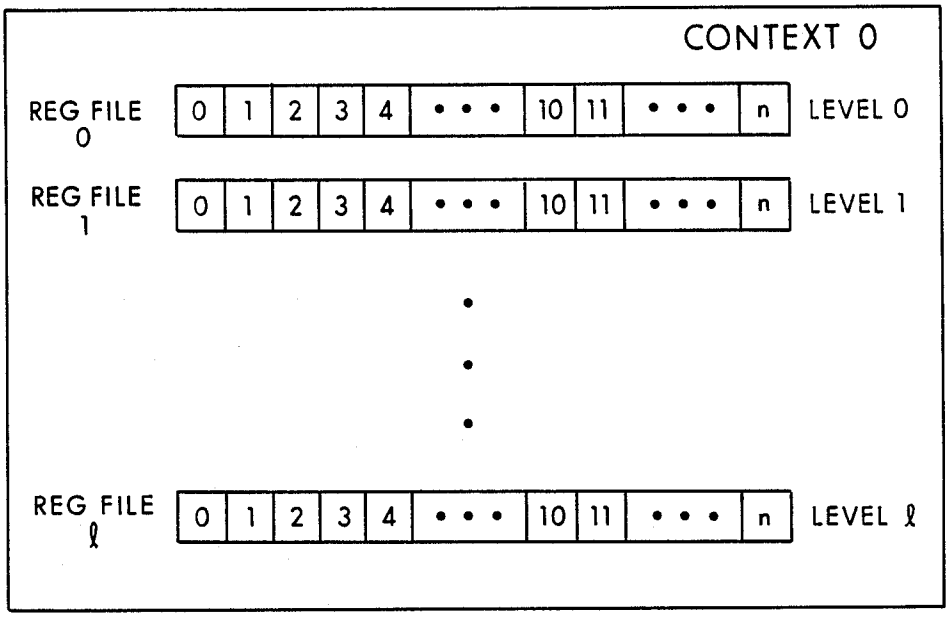

CONTEXT 0
REG FILE 0
REG FILE 1
REG FILE ℓ
0 1 2 3 4 • • • 10 11 • • • n
LEVEL 0
LEVEL 1
LEVEL ℓ

Fig. 13

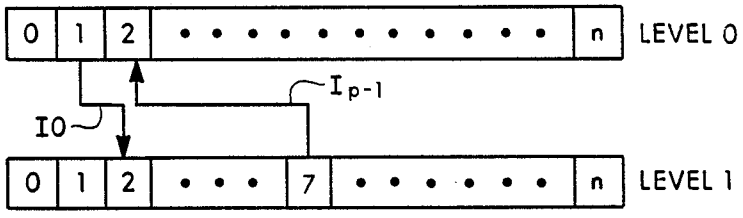

0 1 2 • • • • • • • • • • n
LEVEL 0
I_{p-1}
I0
0 1 2 • • • 7 • • • • • • n
LEVEL 1

Fig. 14

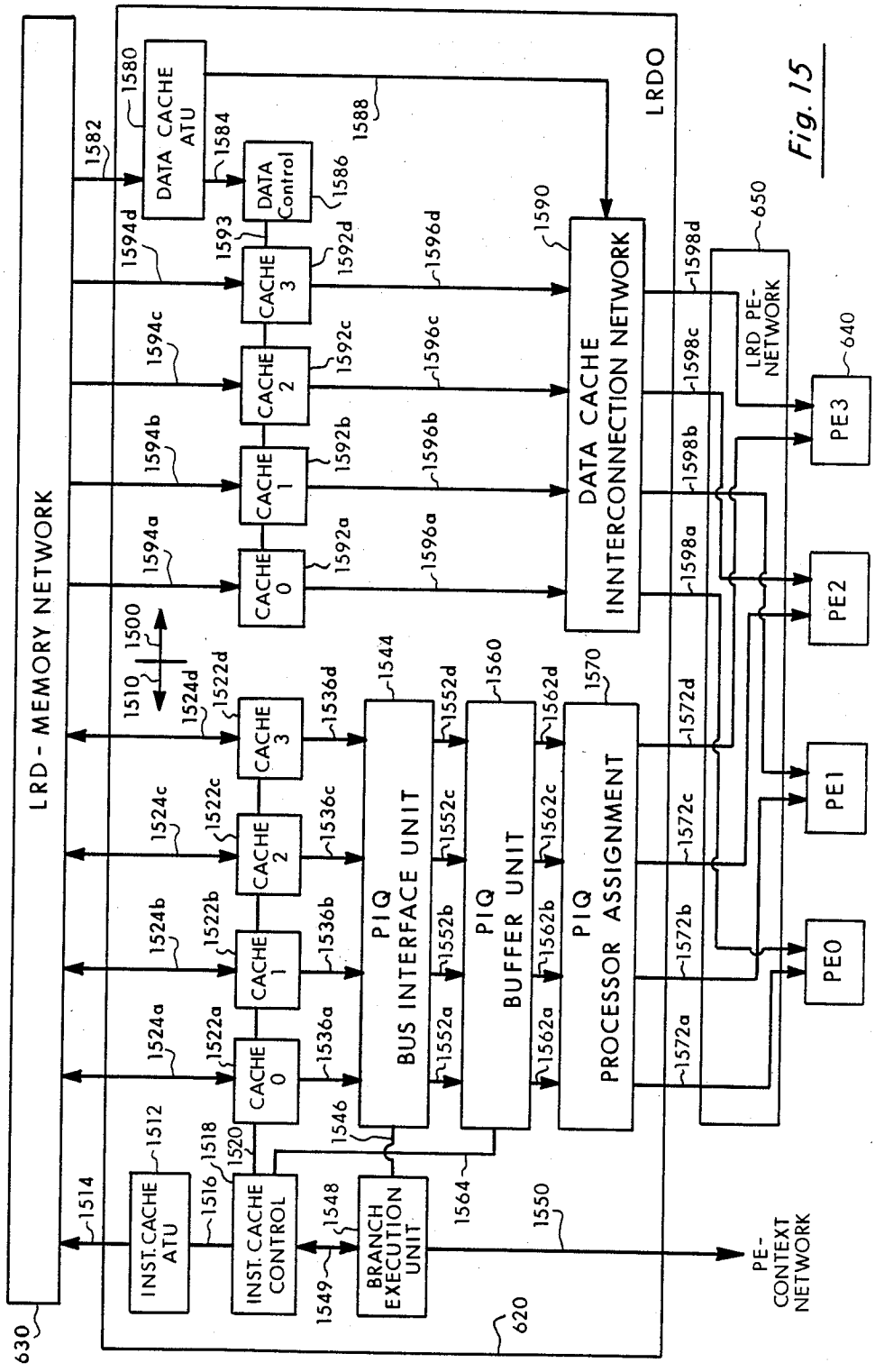

LRD - MEMORY NETWORK
630
1582
1580
DATA CACHE ATU
1584
DATA Control
1586
1588
LRDO
1590
DATA CACHE INNTERCONNECTION NETWORK
1598d
1598c
1598b
1598a
650
LRD PE-NETWORK
640
PE3
PE2
PE1
PE0
1594d
1594c
1594b
1594a
1593
1592d
1592c
1592b
1592a
CACHE 3
CACHE 2
CACHE 1
CACHE 0
1596d
1596c
1596b
1596a
1500
1510
1524d
1524c
1524b
1524a
1522d
1522c
1522b
1522a
1536d
1536c
1536b
1536a
1544
PIQ BUS INTERFACE UNIT
1552d
1552c
1552b
1552a
1560
PIQ BUFFER UNIT
1562d
1562c
1562b
1562a
1570
PIQ PROCESSOR ASSIGNMENT
1572d
1572c
1572b
1572a
1512
1514
1516
1518
1520
INST. CACHE ATU
INST. CACHE CONTROL
1546
1548
1549
BRANCH EXECUTION UNIT
1564
1550
620
PE-CONTEXT NETWORK

Fig. 15

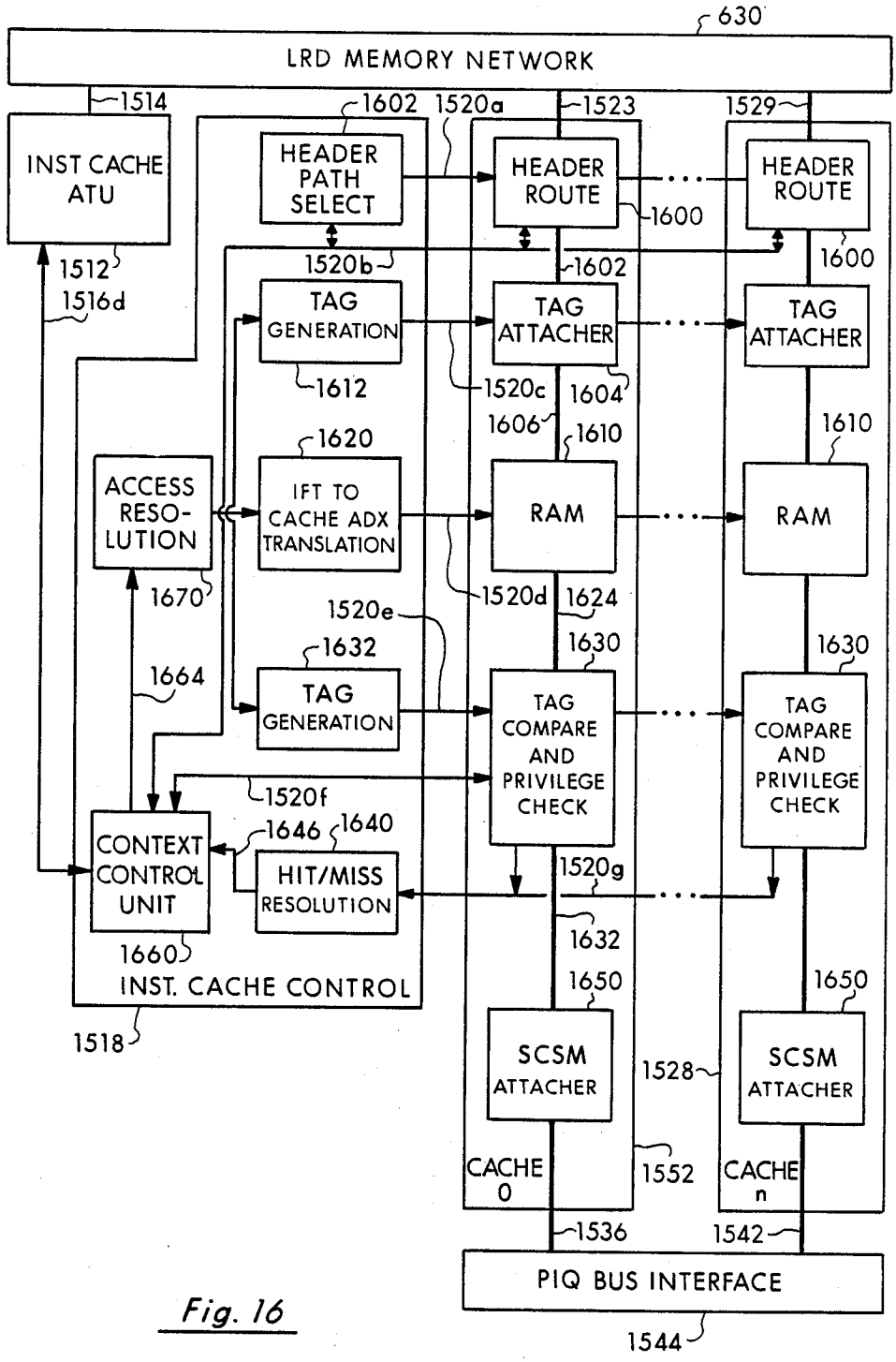

630
LRD MEMORY NETWORK
1514
INST CACHE ATU
1512
1516d
1602
HEADER PATH SELECT
1520b
TAG GENERATION
1612
1620
IFT TO CACHE ADX TRANSLATION
1632
TAG GENERATION
1520e
1664
ACCESS RESO-LUTION
1670
1520f
CONTEXT CONTROL UNIT
1660
1646
1640
HIT/MISS RESOLUTION
INST. CACHE CONTROL
1518
1520a
1523
HEADER ROUTE
1600
1602
TAG ATTACHER
1520c
1604
1606
1610
RAM
1520d
1624
1630
TAG COMPARE AND PRIVILEGE CHECK
1520g
1632
1650
SCSM ATTACHER
CACHE 0
1552
1536
1529
HEADER ROUTE
1600
TAG ATTACHER
1610
RAM
1630
TAG COMPARE AND PRIVILEGE CHECK
1650
SCSM ATTACHER
1528
CACHE n
1542
PIQ BUS INTERFACE
1544

Fig. 16